ERC32 Tools

SPACEBEL INFORMATIQUE s.a.



# ERC32 Products Day
# SPACEBEL Informatique Tools

Commercial Availability

☞ *Target Simulator*

Analyser

Scheduler Simulator

Fernand Quartier

ERC32 Target Simulator SPARC

ERC32 Scheduler Simulator

ERC32 Schedulability Analyser

# Design Goals (1)



- Accurate but fast parallel simulation of an ERC32 based board with following components;
  - Integer Unit (IU), Floating Point Unit (FPU), Memory and Error Controller (MEC), ADA Task Accelerator (ATAC)
  - timers, UART's, DMA, interrupts
  - interfaces for register transfer level and data transfer level I/O simulation, OS emulation
- Easy interface with external products:
  - Thomson Software Product's (TSP) ADA cross debugger
  - external simulators
  - scripts and test languages

# Design Goals (2)



- Versatile *productivity* tool for increased use during the system life cycle (proto's till maintenance)
  - speed and accuracy (up to 1 % real time, 1 cycle resolution)
  - In Circuit Emulator (ICE) capability
  - Tcl/Tk based interface: command line, scripts, windows and pipe interfaces
  - user customisation of board characteristics, EPROM, user interfaces and external interfaces
  - very fast loading of most industry standard formats
  - statistics on word and byte accesses, fetches, loads and stores, IU & FPU activity, power down (idle)

ERC32 Tools

SPACEBEL INFORMATIQUE s.a.

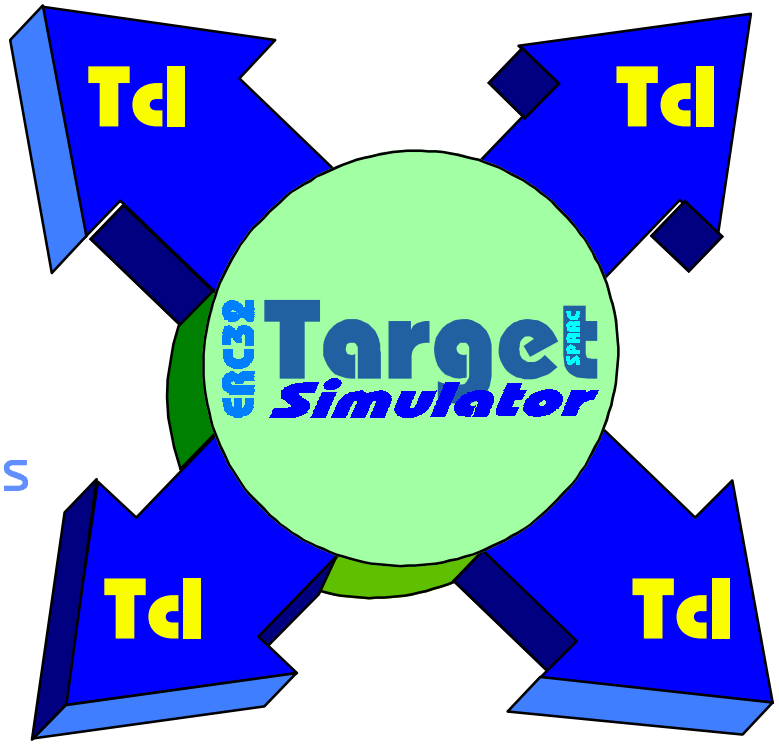
# Versatility

## Windows

- Command Window
- Scoreboard Window
- Code Window
- IO Environment

## User Interfaces

- TSP's ADA Cross Compiler or own debugger
- Scoreboard Window
- Code Window
- IO Environment




## External Interfaces

- I/O Simulator
- OS Emulator
- External debuggers
- Test languages

## User Defined

- Board characteristics
- EPROM
- Scoreboard window
- External interfaces

ERC32 Products Day - Estec - 01/10/96

ERC32 Target Simulator SPARC

ERC32 Scheduler Simulator

ERC32 Schedulability Analyser

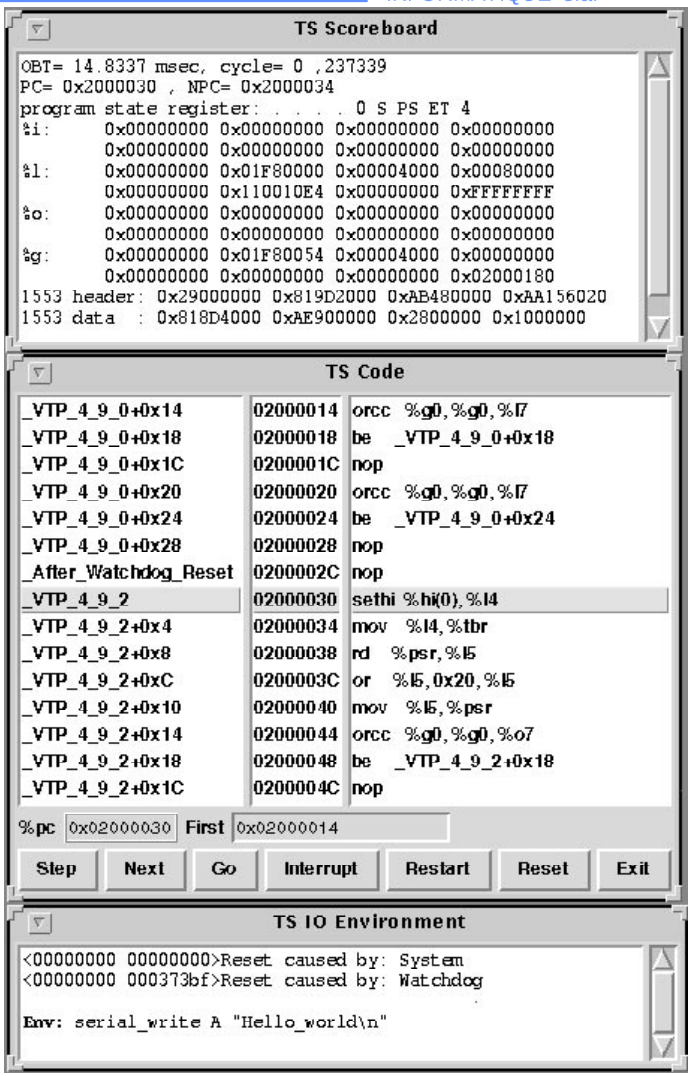
# Debugger (1)



- Supports a.out, ELF, COFF, S_Record and own ASCII dump format
- Generic marker detection system
  - completely non-intrusive: hardware is frozen
  - triggered by markers (not code patches) on a specific address
  - can be further refined with conditions (I.e. if value = constant)
  - triggers breakpoint, start_time, stop_time, tracing, OS_Emulation or IO_Simulation interfaces (combination allowed)
  - in addition, watchpoints can have an address range and access mode (read, write & execute)

# Debugger (2)

- Command Window (not shown)
  - command line interface with Tcl enhancements (history, recursive, expression evaluation ...)
- Scoreboard Window (top)
  - refreshed at each break (step, next)
  - user definable - can contain user Tcl procedures
- Code Window where the PC is
- IO Environment Window
  - all IO, UART and interrupt output and input



# Debugger (3) ICE Capability



- Inject errors and interrupts as specified by the user
- Set breakpoints on memory accesses
  - not necessarily executed code
  - can be refined with conditions
  - can be complemented with global conditions: error, interrupt, stop, log
- Can step through interrupt routines
- Break freezes context (can shorten long tests)
  - HW & SW context can be saved on disk
  - HW & SW context can be restored from disk

# User Configuration



- Board: clock speed, memory banks, sizes and speeds
- EPROM --> easy to integrate in SW production
- Scoreboard Window content for improved efficiency
- External interfaces
- Internal user defined Tcl procedures for OS_Emulation and I/O simulation
  - read at startup
  - execute internally (fast)
  - can interface with the external world through pipes